Doc. No. 9902-A002-0001

9 October 1999

SCIENTIFIC AND TECHNICAL REPORT

(INTERIM)

VISUALIZATION OF WEAPONS EFFECTS FOR TRAINING AND TEST & EVALUATION

TECHNICAL REPORT - STUDY/SERVICES

CDRL SEQUENCE NO. A002

9 October 1999

Prepared by:
Brian Cahill
Principal Investigator

DUAL INCORPORATED
30 Skyline Drive
Lake Mary, Florida 32746-6238

Contract Number:	N00421-99-C-1383
Contract Value:	$99,982.54 (w/option)
Contract Award:	Competitive
Sponsor POC:	NAVAIRSYSCOMHQ PMA 205 Attn: Cdr. Michael Holmes, SBIR Coordinator, Bldg. 2272, Suite 345 47123 Buse Road Unit IPT Patuxent River, MD 20670-1547

TABLE OF CONTENTS

I. INTRODUCTION

This interim final report is prepared under Contract N00421-99-C-1383 to document and summarize technical accomplishments for the first four months of the program. The Virtual Environment for Weapons Effects Visualization (VEWEV) is a Phase I effort of the Small Business Innovation Research (SBIR) Topic N99-062 of DoD Program Solicitation 99.1. The six-month Phase I contract (with three-month option) was awarded on 8 June 1999. Further guidance was received from the Contracting Officer's Technical Representative (COTR) during a 25 June orientation meeting. The University of Central Florida (UCF) was given a verbal go-ahead by Dual, Incorporated (Dual) to begin work on the project, and was formally brought under subcontract on 6 August.

The goals of the project are to: (1) demonstrate that ballistic effects can be visualized, and (2) demonstrate that visualization of ballistics effects can be embedded in a training device. Goal 1 is the primary goal of the Phase I effort, and is nearly fully realized, using representative models of a rotary-wing air vehicle and a light ground vehicle. Goal 2 will be the main focus of Phase II.

II. INTERIM RESULTS

A. Finite Element Modeling & Analysis

Pre-Processor: FEMB

The Finite Element Model Builder (FEMB) is used as a pre-processor for LS-DYNA 3D. Through FEMB the user defines points, lines, surfaces, nodes and elements, generating the desired structure. Once this task is achieved, still inside FEMB, the user can define the boundary conditions and apply loads to the geometry. Unfortunately, there are only a small amount of materials and properties that are currently supported by FEMB, which means that the user has to go through a user interface whenever they need to use a non-supported element.

The helicopters presented in this report were originally obtained as Drawing Exchange Format (DXF) files, as represented in Figure 1 (see Appendix A for figures). Many different aircraft structures (e.g., B-17, Bell, Cobra, Commanche, MIG, DC-10, F-14, F-16, HIND24, MK9HAWK) were obtained from a public URL on the Internet. All of these files were originally in DXF format. LS-DYNA 3D is only able to read lines from DXF files, due to the translator currently built into the software. Unfortunately, the files downloaded from the Internet represent models in terms of areas and surfaces, which cannot be read by the LS-DYNA 3D translator software. Fortunately, the Initial Graphics Exchange Specification (IGES) translator in LS-DYNA 3D is capable of reading the model elements of interest. For that reason, the files were translated into IGES files inside AutoCAD before being read into LS-DYNA 3D.

While importing files, LS-DYNA 3D prompts for missing information/data and reports errors. Fortunately, no error messages were encountered during this exercise, meaning that the whole files were translated properly inside LS-DYNA 3D. The IGES files were read by FEMB into FMB files.

Once the geometry was inside LS-DYNA 3D, it became obvious that only points, surfaces and volumes were defined. In order to transform the IGES files into a finite element model, nodes and elements would have to be created. Unfortunately, the package is not able to automatically create those entities from the available point, surface, and volume information. Therefore, nodes and elements were created manually inside LS-DYNA 3D. In order to ensure the connectivity between the elements, minor adjustments were performed in the given files. Figure 2 represents only the outside structure of a military helicopter. The internal structure and the rotors were not included in this simulation. At this point of the project, however, those missing parts represent no significant difference; they would only represent more elements to the simulation. Furthermore, the outside structure plays an important role to any military vehicle. It has to give an appropriate shielding effect and it has to be as light as possible. For that reason, the initial simulations are all performed using the structure shown in Figure 2. Once the files were transformed into LS-DYNA 3D executable files (DYN format), the impacts, crashes and explosion simulations were all implemented through the user interface.

The first simulation performed in the software was the crash of the helicopter structure against a mountain at 75 mph. This is a realistic scenario and it has happened already due to accidents or due to combat maneuvers. The mountain is simulated here as a rigid wall. The helicopter hits the mountain at an angle of 30 degrees. Figure 3 depicts the structure ready for analysis with all the boundary conditions and the initial conditions already defined by the user. The sliding surfaces are also defined as well. The crash angle and position chosen was one of the worst possibilities for the pilot.

The numerical solution of the crash simulation is shown in Figure 4. As was obviously expected, the mountain (rigid wall) does not move or deform. The helicopter, however, crashes and its front structure is thrown inside, in the direction of the pilot. This simulation is extremely important, as it enables the test of many different materials that can be used as an aircraft shield.

The next simulation performed in LS-DYNA 3D was a projectile impact. An Armor Piercing Fin Stabilized Discharging Sabot (APFSDS) projectile was chosen. Although this type of ammunition is not typically used against helicopters, it is well known to have maximum penetration properties. That is the reason why it is used against main battle tanks. However, for the sake of illustration, this simulation was performed in LS-DYNA 3D. The APFSDS projectile impacts the structure at 4500 feet per second. As was expected, a full penetration was observed as a result of this simulation (see Figure 5).

Since this kind of simulation cannot be performed in real-time for purposes of visualization, the idea of creating a library of possible zones of ballistic impact was conceived. Once the user defines a zone, the closest damage result would be shown, according to hit probability principles. This will considerably enhance the time response of the system. As a starting point, eight simulations are performed according to the octants shown in Figure 6.

Figure 7 shows the result of the fragments hit of a 30mm ammunition (from a BOFORS Cannon) that exploded in the 3rd octant of the helicopter model. Probabilistic distribution indicated that two fragments hit the helicopter structure with enough power to produce damage. The type of fuse used in this case was a proximity fuse, which detonates when it gets close to a metallic structure. At this point, simulations for all eight octants have been performed in LS-DYNA 3D.

As stated in the goals of the project, a light ground vehicle is also a representative model of interest. The truck shown in Figure 8 is a light truck that could be used to transport basic material and/or a small number of people. The hit shown is a single impact due to .50" projectile ammunition. A more realistic scenario due to round shots will also be performed with LS-DYNA 3D.

Other truck models are also available. They are the most complete models that are currently being used (over 65000 elements – beam, shell and solid). However, due to computing hardware limitations, the solution for the complete models cannot yet be solved. To alleviate this problem, the simulations are analyzed by regions (impact zones).

Currently, efforts are being made to come up with a more realistic helicopter to be used for the next steps of the project. The assault helicopter COBRA was chosen. Figure 9 shows the finite element model of this structure, which is currently being transformed into an executable LS-DYNA 3D file.

Analysis: LS-DYNA 3D

LS-DYNA 3D is a finite element impact analysis tool used to perform the numerical simulations of the current work. Once the executable DYN file is ready for analysis, the program LS-DYNA 3D is called to perform the simulation. LS-DYNA 3D is a vectorized explicit three-dimensional tool for analyzing the large deformation dynamic response of inelastic solids. By invoking special capabilities, interfaces can be rigidly tied to admit variable zoning without the need for transition regions (e.g., triangular elements connected to quadrilateral elements).

The time step (which can be modified by the user) is automatically chosen as a function of the smallest element in the geometry. In order to guarantee stability, the Courant condition is used. So, for every time step, the stress, strain and displacement tensors are updated until the total time determined by the user is reached.

Spatial discretization can be achieved by the use of eight node solid elements, two node beam elements, membrane elements, discrete elements and rigid bodies. The equations of motion are integrated explicitly in time by the central difference method. The code also contains a contact-impact algorithm that permits gaps and sliding along material interfaces. Figure 10 represents sliding surfaces for the helicopter crash. LS-DYNA 3D currently contains 39 material models and 10 equations of state to cover a wide range of material behavior. Additional models are available in the code EPIC II.

LS-DYNA 3D has gone through extensive development since its inception in 1976. It has been improved and made more efficient, and new features have been added to meet the needs of impact analysis.

The 1982 version of LS-DYNA 3D accepted material input directly through a user interface. The new organization in the code was such that the user could implement his/her own material model. That offers the possibility of developing equations of state and constitutive models of great complexity. Also, using the user interface, modifications are allowed in the existing models. The majority of the results presented in this report have been obtained by altering existing models built in the pre-processor FEMB through the user interface.

LS-DYNA 3D allows the output of a great variety of variables. Of great importance here at this point of visualization are the displacements and stresses for the nodes and elements respectively. So, the files for the nodes (NODOUT) and the files for the elements (ELOUT) are both flagged to be stored before execution begins. The information contained in these files is used to perform the visualization in VRML.

Explosive Analysis

LS-DYNA 3D has a high velocity burning material already built in. The introduction of Equations of State (EOS) is allowed. The material is supported for solid elements and requires a contact interface algorithm in order to properly transfer the release of chemical energy.

Projectile Impact Analysis

Another type of loading of interest for this project is projectile impact analysis. In a different range from an explosive loading in terms of velocity, energy transfer and area of action, projectile impact can severely damage a structure. LS-DYNA 3D allows the development of any type of projectile (rigid, deformable, eroding, etc).

Post-Processor: FEMB

FEMB is also used as a post-processor for LS-DYNA 3D. Once the numerical simulation is performed, all results that were flagged before the execution can be visualized into FEMB. Stresses, strains, failures, element deletion, energy curves, etc. represent the available output response that can be read into FEMB. In reality Figures 4,5,7 and 8 are numerical results captured from FEMB. Figure 11 represents a rigid wall force graph for the crash scenario.

B. Format Translation

Dyn2VRML is a fully-integrated, user-friendly, Windows-based application that translates the standard ASCII output from the dynamic analysis code LS-DYNA 3D into the standard VRML1.0 format.

The core translation subroutines are written in FORTRAN90 using double precision dynamically allocatable arrays for memory optimization. These core subroutines are compiled into Dynamic Link Libraries (DLLs) to be used by the Windows-driven graphical interface, currently in development. Currently, *Dyn2VRML* works as a console application. The addition of graphical interface features will be one of the points of focus for the remainder of Phase I of the project, along with the embedding of the VRML browser, to make a fully integrated tool.

Main Program

The main program of *Dyn2VRML* structures the translation process into several steps. These steps are described in the following program design language (PDL):

```
BEGIN PDL "MAIN PROGRAM"
     DECLARE dynamic arrays for the translation process.
     (To optimize use of memory, arrays are allocated into
     memory before the subroutine that uses them and are
     deallocated once they are no longer used.)

     OPEN the geometry information output file of LS-DYNA
     3D, which has the extension OTF.

     INPUT the essential information of the model such as
     the number of materials, number of nodes, and number
     of elements from the OTF file.

     CALL the subroutine that reads the geometric
     configuration.
```

```
        CREATE a file to store the initial VRML translation.

        CALL the subroutine that transforms the nodal
        identification into an ascending number sequence.

        CALL the subroutine that filters the nodes of 8-noded
        brick elements that lie inside the body and stores the
        external (visible) nodes.

        CALL the subroutine that generates the connectivity of
        surface elements from brick element filtering.

        CALL the subroutine that translates the information
        into an initial VRML file that contains the model
        geometry and the material composition.

        OPEN the deformation file for the model output by LS-
        DYNA 3D.

        CALL the subroutine that reads the nodal displacement
        (deformation).

        OPEN the stress distribution file for the model output
        by LS-DYNA 3D.

        CREATE a file to store the final VRML translation.

        CALL the subroutine that translates the information
        into a final VRML file that contains the model
        geometry, deformation, and stress distribution.

    END PDL "MAIN PROGRAM"
```

Each subroutine called by the main program is detailed below.

InputGeometry Subroutine

The subroutine *InputGeometry* of *Dyn2VRML* reads the initial location of the geometric nodes of the model sequentially as well as the connectivity matrices for the four types of elements: (1) 8-noded brick elements, (2) 3-noded beam elements, (3) 4-noded shell elements, and (4) 8-noded thick shell elements.

```
    BEGIN PDL "InputGeometry"

        DECLARE the arguments.

        READ the nodal location.

        READ the brick elements connectivity and material.

        READ the beam elements connectivity and material.

        READ the shell elements connectivity and material.

        READ the thick shell elements connectivity and
        material.
    END PDL "InputGeometry"
```

CorrectConnectivity Subroutine

The subroutine *CorrectConnectivity* of *Dyn2VRML* transforms the nodal identification numbers into an ascending sequence to ease the future reference from connectivity matrices.

```
BEGIN PDL "CorrectConnectivity"
    DECLARE the arguments.
    LOOP over the total number of nodes
        COMPARE if the node number coincides with the node ID.
        If not, LOOP over the total number of elements and
        replaces the node ID for the node number.
    END LOOP
END PDL "CorrectConnectivity"
```

FilterBrickSurface Subroutine

The subroutine *FilterBrickSurface* of *Dyn2VRML* eliminates the nodes of 8-noded brick elements that lie inside the body for memory and rendering optimization, by applying the simple principle that nodes inside the body are shared by more elements than those on the surface. The code only keeps those nodes lying on the surface, which are the only ones needed to visualize a 3-D body.

```
BEGIN PDL "FilterBrickSurface"
    DECLARE the arguments.
    LOOP over all brick elements
        RECORD the nodes that are shared by less than five
        elements.
        COUNT the number of surface elements.
    END LOOP
END PDL "FilterBrickSurface"
```

GenerateBrickSurface Subroutine

Subroutine *GenerateBrickSurface* of *Dyn2VRML* generates the new surface element connectivity from the node filtering of the brick elements.

```
BEGIN PDL "GenerateBrickSurface"
    DECLARE the arguments.
    LOOP over the total number of brick elements
        RECORD the new nodal connectivity for surface elements
        from the old brick elements.
    END LOOP
END PDL "GenerateBrickSurface"
```

OutInitVrml Subroutine

Subroutine *OutInitVrml* of *Dyn2VRML* translates the initial nodal location and material composition into a VRML1.0 file.

```
BEGIN PDL "OutInitVrml"
    DECLARE the arguments.
    COMPUTE the dimensions of the model.
    SET the position and stepping of the camera proportional to
    the biggest dimension of the model.
    SET the orientation of the camera perpendicular to the
    biggest dimension of the model.
    CREATE a directional light diagonal to the initial
    perspective.
    WRITE the brick elements outer surfaces with its
    corresponding material color in shades of gray using four
    coordinates and indexed face sets.
    WRITE the beam elements with its corresponding material
    color in shades of gray using three coordinates and indexed
    line sets.
    WRITE the shell elements with its corresponding material
    color in shades of gray using four coordinates and indexed
    face sets of both sides of the shell.
    WRITE the thick shell elements with its corresponding
    material color in shades of gray using eight coordinates and
    indexed face sets.
END PDL "OutInitVrml"
```

InputDeformation Subroutine

Subroutine *InputDeformation* of *Dyn2VRML* reads the nodal displacement from the ASCII output file *nodout* from the LS-DYNA 3D analysis.

```
BEGIN PDL "InputDeformation"
    DECLARE the arguments.
    READ the nodal displacement sequentially.
    ADD the displacement to the initial nodal positions to form
    the deformed model.
END PDL "InputDeformation"
```

InputStress Subroutine

The subroutine *InputStress* of *Dyn2VRML* reads the stress distribution over the elements from the ASCII file *elout* from the LS-DYNA 3D analysis.

```
BEGIN PDL "InputStress"
    DECLARE the arguments.
    READ the stress tensor for the brick elements outer surfaces
    and COMPUTE the von Mises stress.
    READ the stress tensor for the beam elements and COMPUTE the
    von Mises stress.
    READ the stress tensor for the shell elements and COMPUTE
    the von Mises stress.
    READ the stress tensor for the thick shell elements and
    COMPUTE the von Mises stress.
END PDL "InputStress"
```

OutFinalVrml Subroutine

Subroutine *OutFinalVrml* of *Dyn2VRML* translates the final deformed nodal location and von Mises stress distribution into a VRML1.0 file.

```
BEGIN PDL "OutFinalVrml"
    DECLARE the arguments.
    COMPUTE the dimensions of the model.
    SET the position and stepping of the camera proportional to
    the biggest dimension of the model.
    SET the orientation of the camera perpendicular to the
    biggest dimension of the model.
    CREATE a directional light diagonal to the initial
```

```
        perspective.
        WRITE the brick elements outer surfaces with its
        corresponding stress color by a quadratic RGB composition
        using four coordinates and indexed face sets.
        WRITE the beam elements with its corresponding material
        color by a quadratic RGB composition using three coordinates
        and indexed line sets.
        WRITE the shell elements with its corresponding material
        color by a quadratic RGB composition using four coordinates
        and indexed face sets of both sides of the shell.
        WRITE the thick shell elements with its corresponding
        material color by a quadratic RGB composition using eight
        coordinates and indexed face sets.
    END PDL "OutFinalVrml"
```

Complete source code listings will be provided in the final submission of this document.

C. Visualization

Using the current console version of *Dyn2VRML,* a series of preliminary analysis models (described in Section II.A) have been translated. Excellent results have been obtained with the translator, generating VRML files with good resolution and fast rendering. A brief description of some of the preliminary visualizations (found in Appendix A) follows.

- Cylindrical beam under compression (Beam01.otf). This model displays a quarter portion of a cylindrical beam under symmetrical compression. The VRML model provides a clear distribution of the von Mises stress concentration. (Figure 12)
- Ford Fiesta under mine blast (FiestaBlast01.otf). This detailed model contains brick, beam, and shell elements and was used to test the translator with multi-element configuration. The car drives over a mine with the front left tire and receives the blast from below. (Figure 13)
- Helicopter model under mountain impact (ChopperImpact01.otf). This model depicts a coarse discretization for a military helicopter crashing into a solid wall at a constant speed. (Figure 14)
- Helicopter model under projectile blast (ChopperBlast01.otf). This is the same geometric model as the previous example, but in this case a projectile impacts and goes through the side of the chopper. (Figure 15)
- C2500 light truck under missile blast (C2500Blast01.otf). This is a detailed model of a light truck, which contains all different types of elements. The analysis is performed with a projectile impact to one of the sides. (Figure 16)

Representative examples of the VRML models generated by the translator from LS-DYNA 3D output can be viewed at http://home.mpinet.net/edivo/dual. See http://cws.internet.com for shareware VRML browsers available for download. We recommend the *Cosmo* player from SGI for viewing VEWEV files.

III. FUTURE PLANS

A. Remainder of Phase I

In the time remaining for Phase I, the selected models will be further refined. An integrated Graphical User Interface (GUI) will be developed for the translation/visualization utility. A demonstration database of Weapon Effects Files (WEF) will be built.

The VEWEV technology will be demonstrated at the Interservice/Industry Training, Simulation and Education Conference (I/ITSEC). I/ITSEC will be held the week of 29 November at the Orange County Convention Center in Orlando, Florida. VEWEV will be demonstrated in the conference's "Technology Showcase" at 9:00am ET on Thursday, 2 December 1999. See. http://www iitsec.org for conference details.

In addition, UCF is taking the lead with Dual input in authoring a paper about the project for an upcoming LS-DYNA conference. More details will be provided in the monthly reports as the conference draws nearer.

B. Phase I Option

In the three month Phase I Option, further applications research for the VEWEV will be conducted, the WEF database will be expanded, and a demonstration video will be produced.

C. Phase II

The secondary goal of the Phase I effort becomes the primary goal of Phase II: Demonstrate that visualization of ballistics effects can be embedded in a training device. To that end, Dual has been seeking a Government sponsor with training applications suitable to benefit from and demonstrate the utility of this technology. A promising contact has been made within the U.S. Army's Simulation, Training and Instrumentation Command (STRICOM). The Program Manager for Instrumentation, Targets and Threat Simulators (PMITTS) has expressed a possible interest in sponsoring a Phase II effort. This interest is preliminary and will be followed up in the coming weeks.

Another key activity of Phase II will be the development of a commercialization strategy for the technology.

IV. CONCLUSION

As stated earlier, the primary goal of the Phase I project is to demonstrate that ballistic effects can be visualized. As evidenced by the data in this report, this goal is nearly fully realized, using representative models of a rotary-wing air vehicle and a light ground vehicle. Tremendous progress has been made to date and will continue to be made throughout the remainder of Phase I and the Phase I Option. Phase II will take the technology from proof of concept to practical use.

Appendix A - Figures

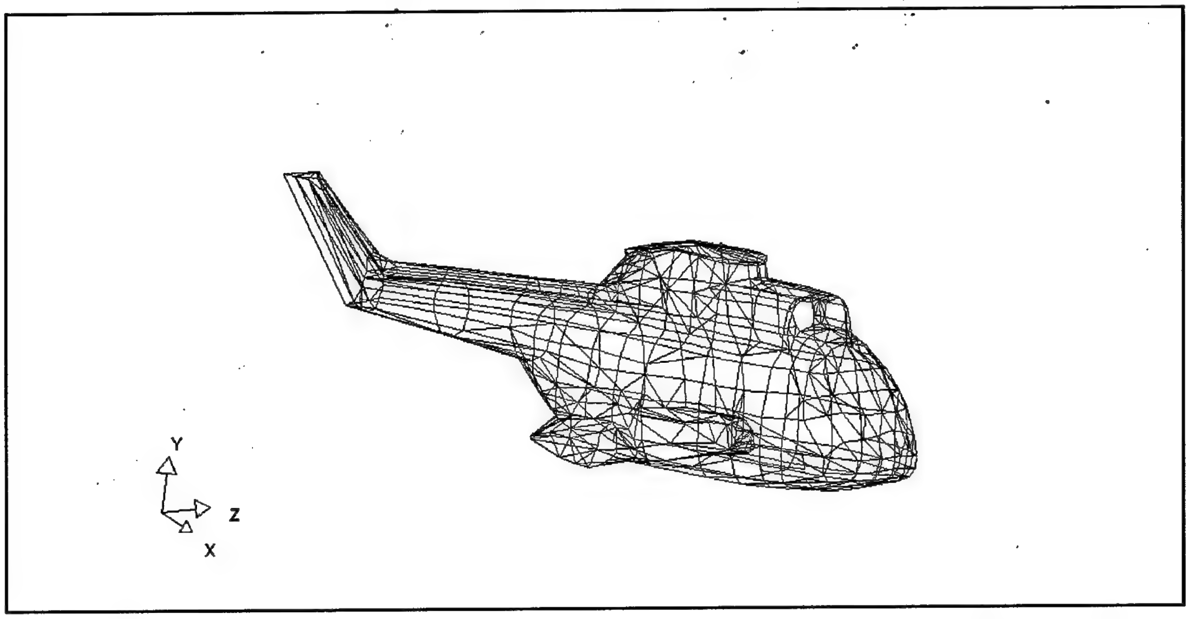

Figure 1. Helicopter Structure Originally Obtained as a DXF File.

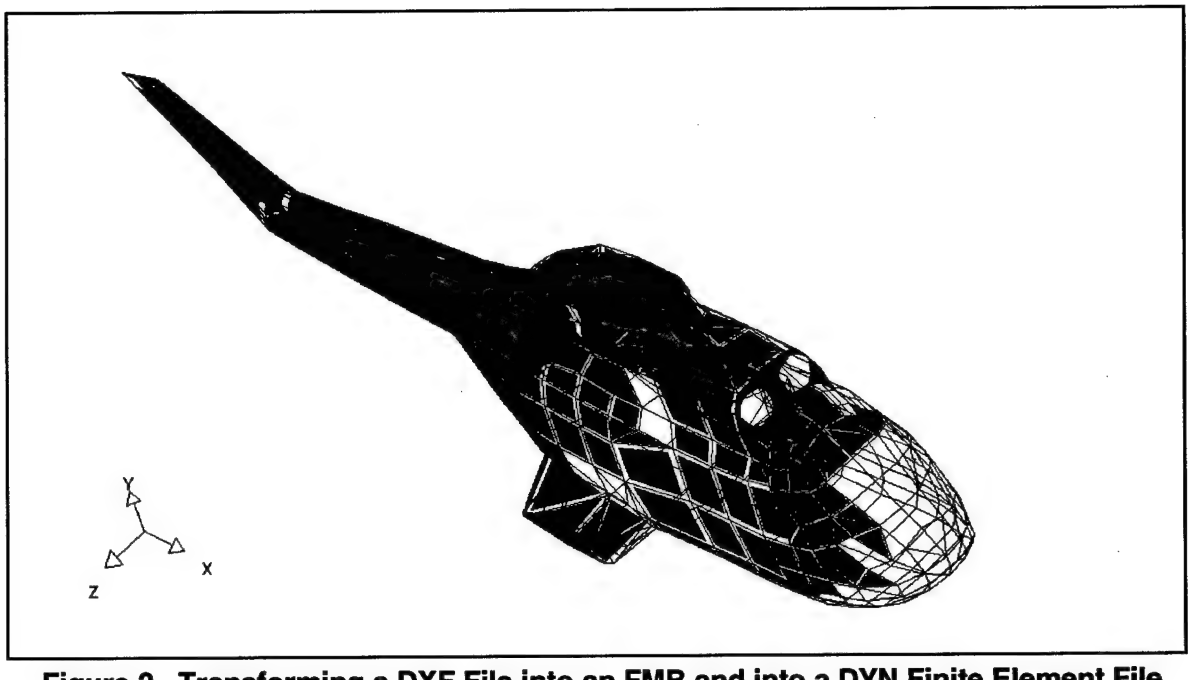

Figure 2. Transforming a DXF File into an FMB and into a DYN Finite Element File.

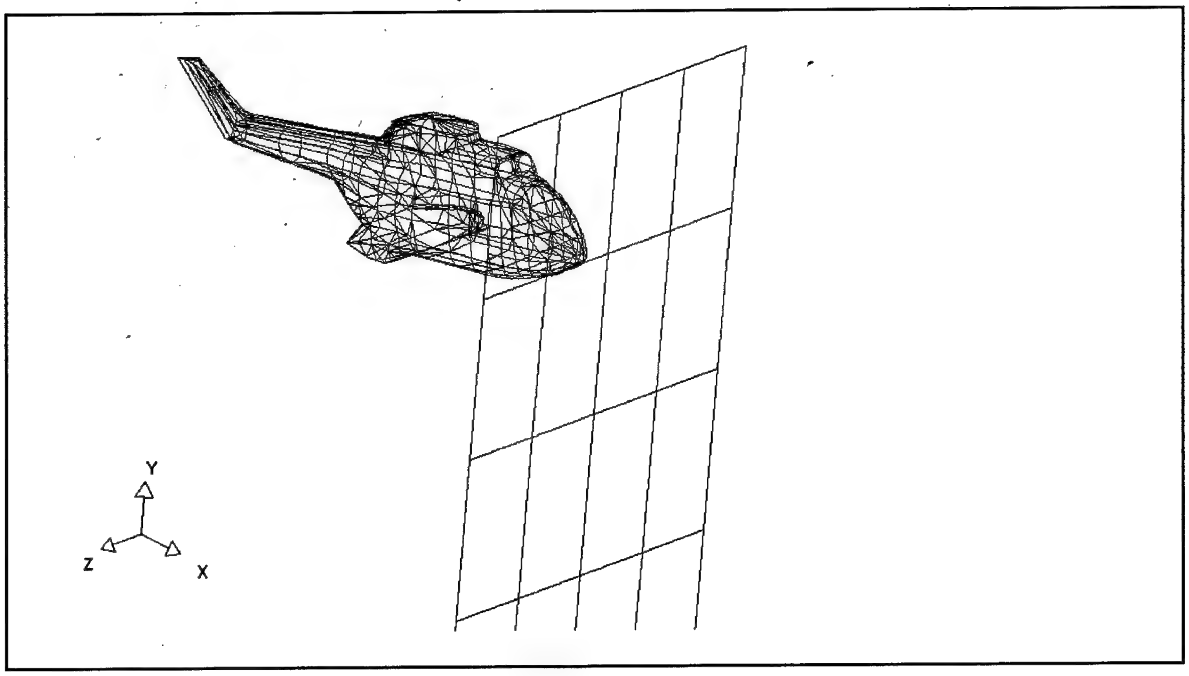

Figure 3. Helicopter Structure against a Rigid Wall, Ready for Analysis.

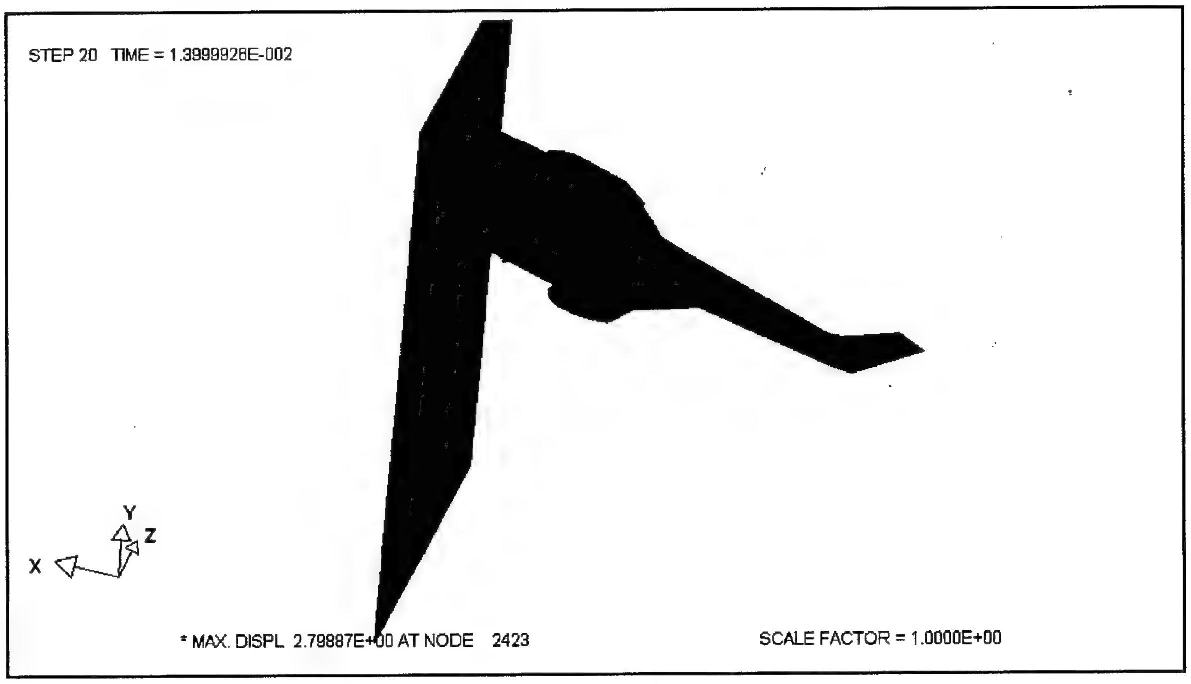
STEP 20 TIME = 1.3999928E-002
Y
Z
X
* MAX. DISPL 2.79887E+00 AT NODE 2423
SCALE FACTOR = 1.0000E+00

Figure 4. Numerical Results of the Helicopter Crash.

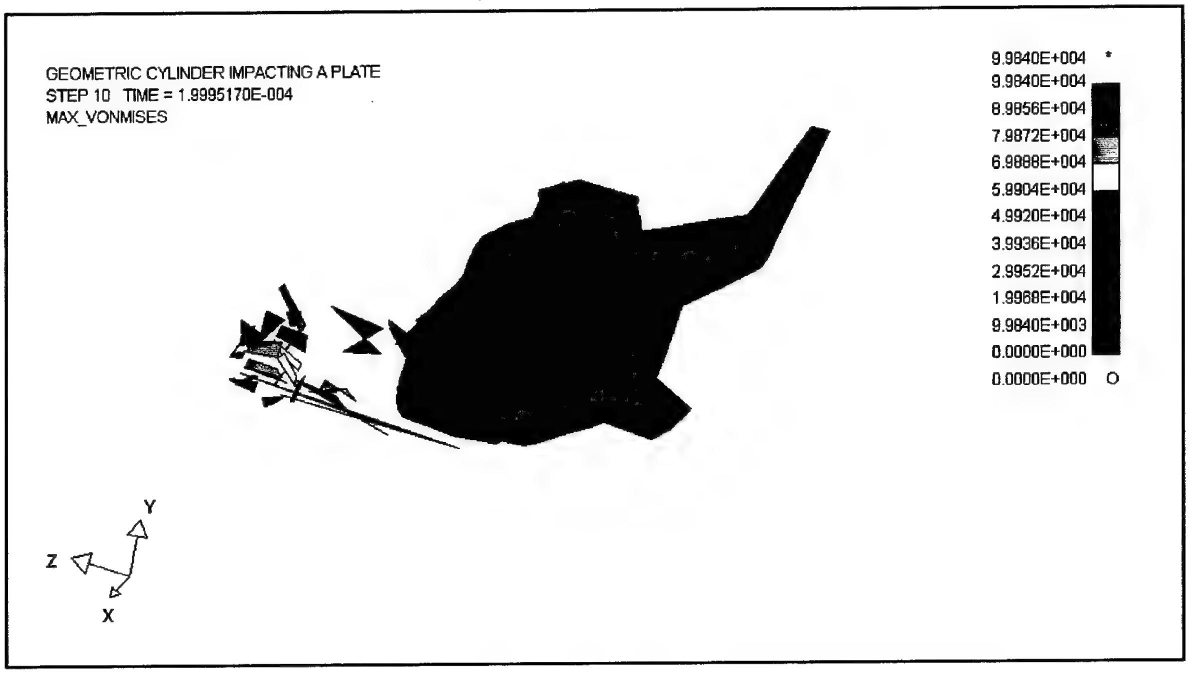

Figure 5. Helicopter Structure Hit by APFSDS Ammunition.

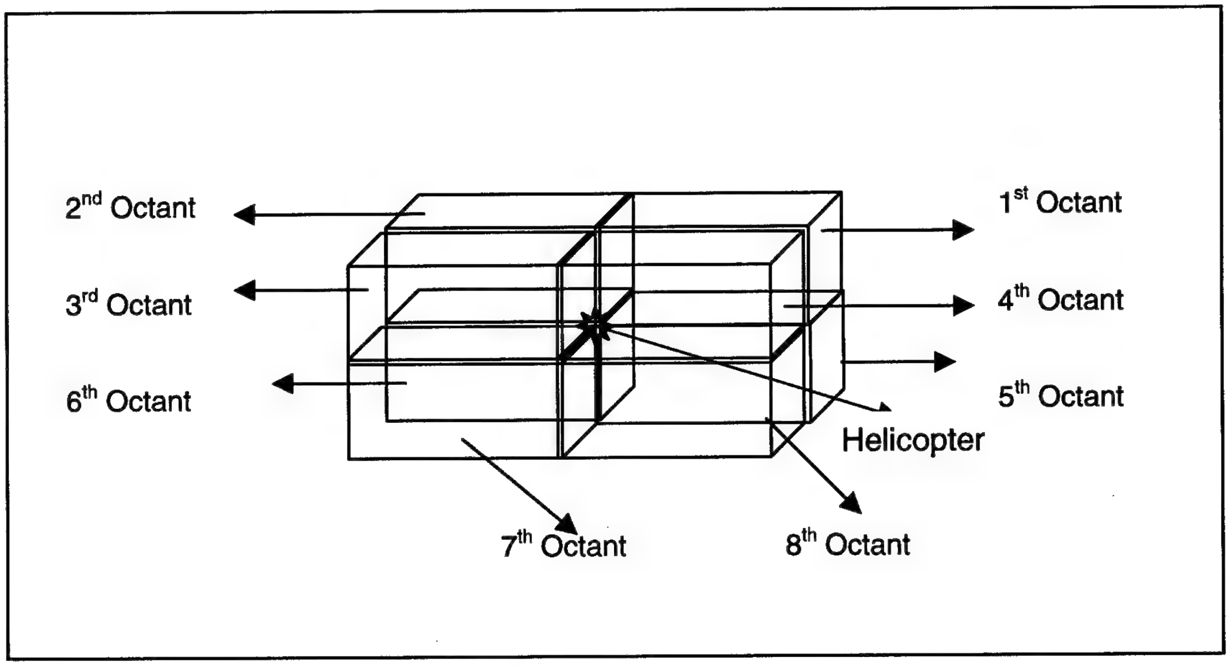

Figure 6. Zones of Ballistic Impact.

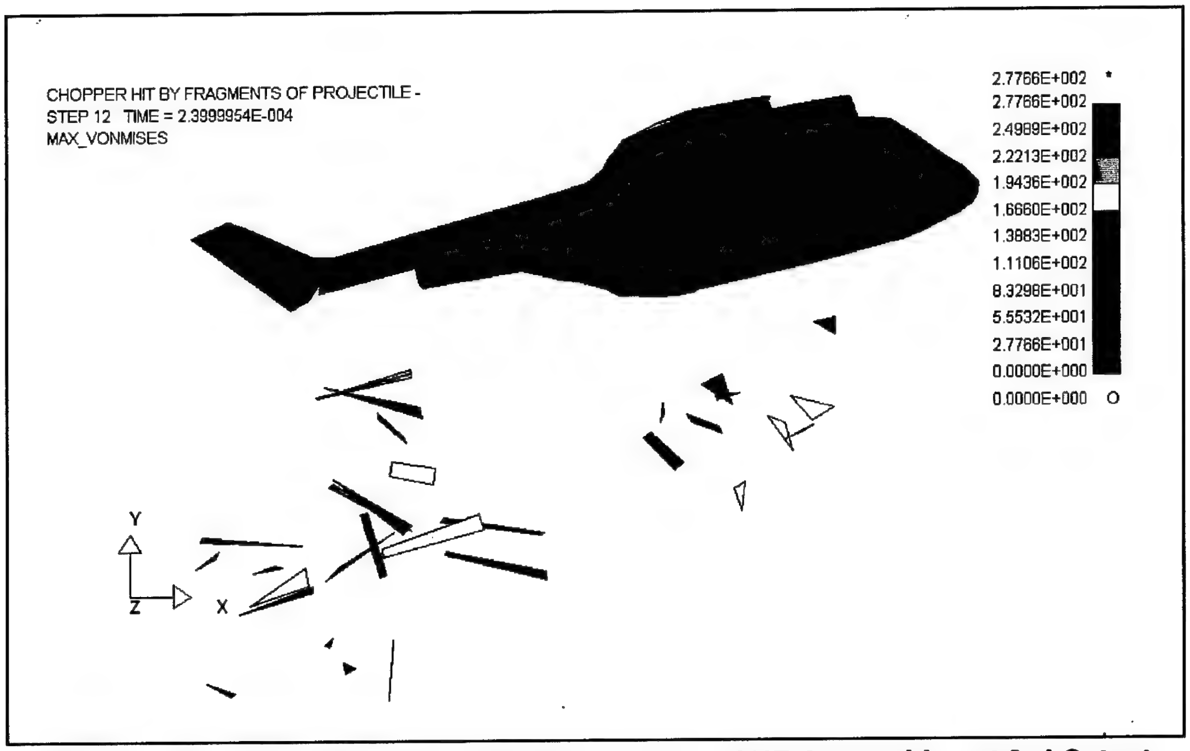

Figure 7. Helicopter Structure Hit by Fragments of HE Ammunition at 3rd Octant.

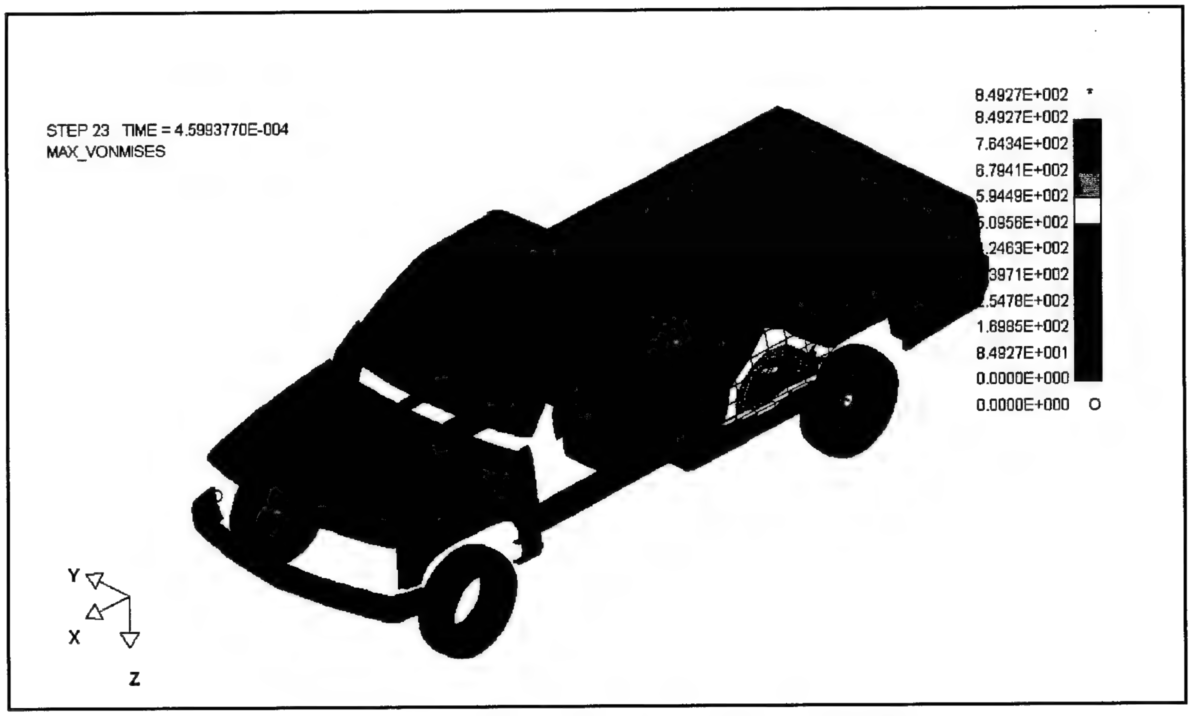

Figure 8. Light Truck Hit by a .50" Projectile.

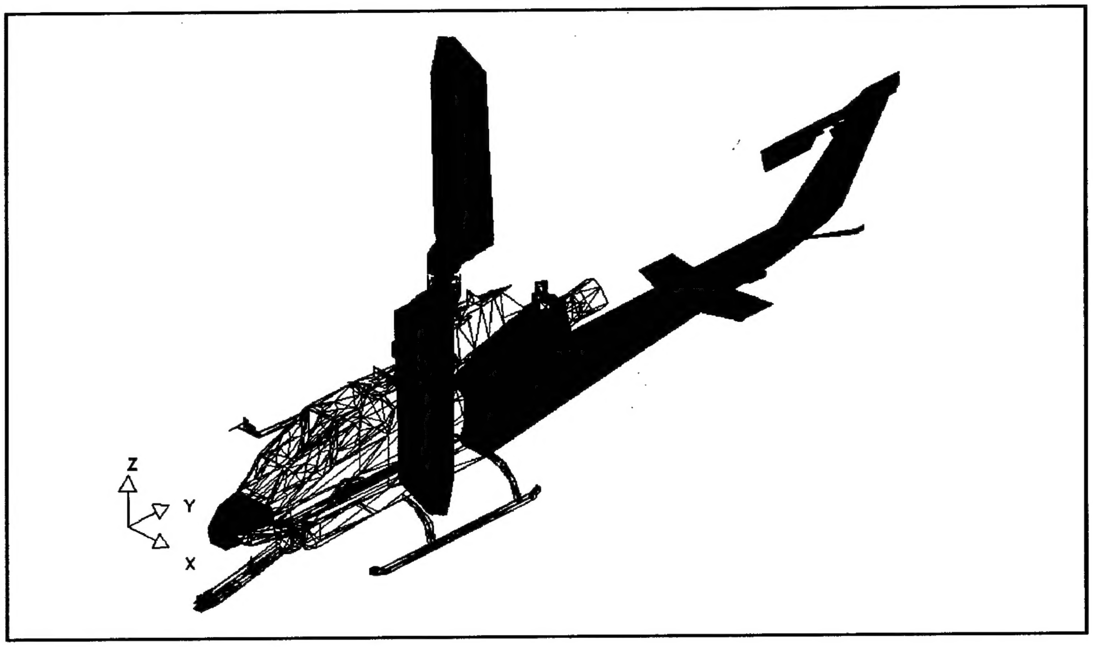

Figure 9. COBRA Helicopter DXF File Being Converted Into DYN File.

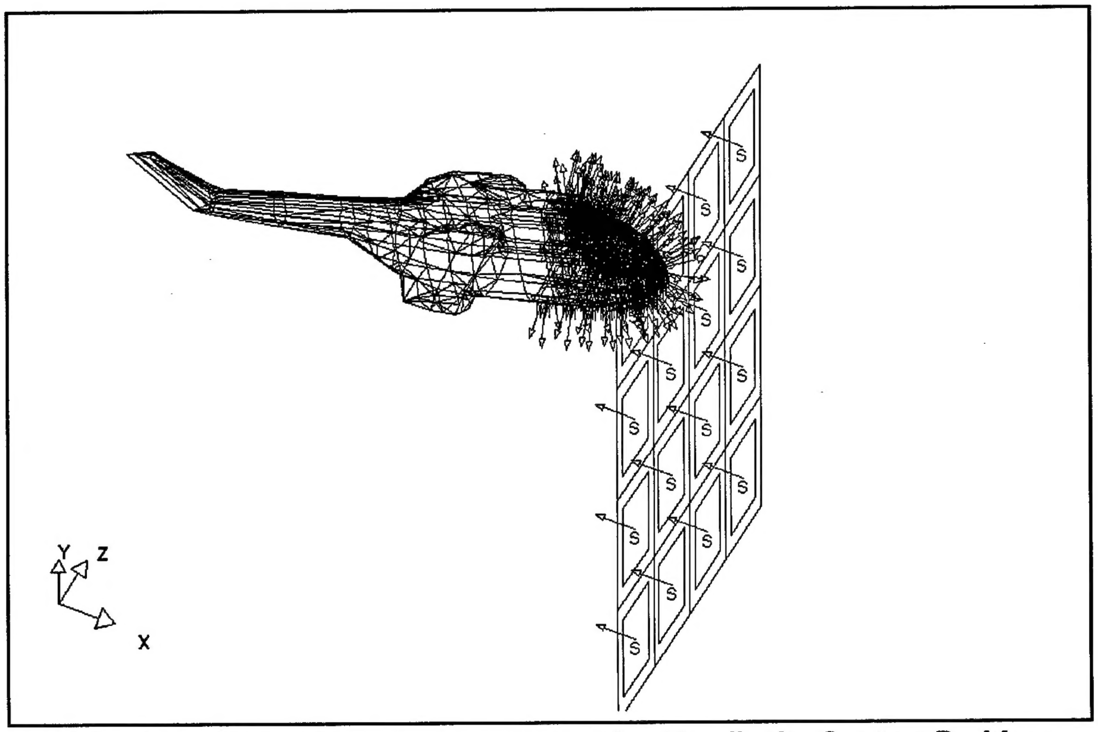

Figure 10. Sliding Lines/Surfaces Created to Handle the Contact Problem.

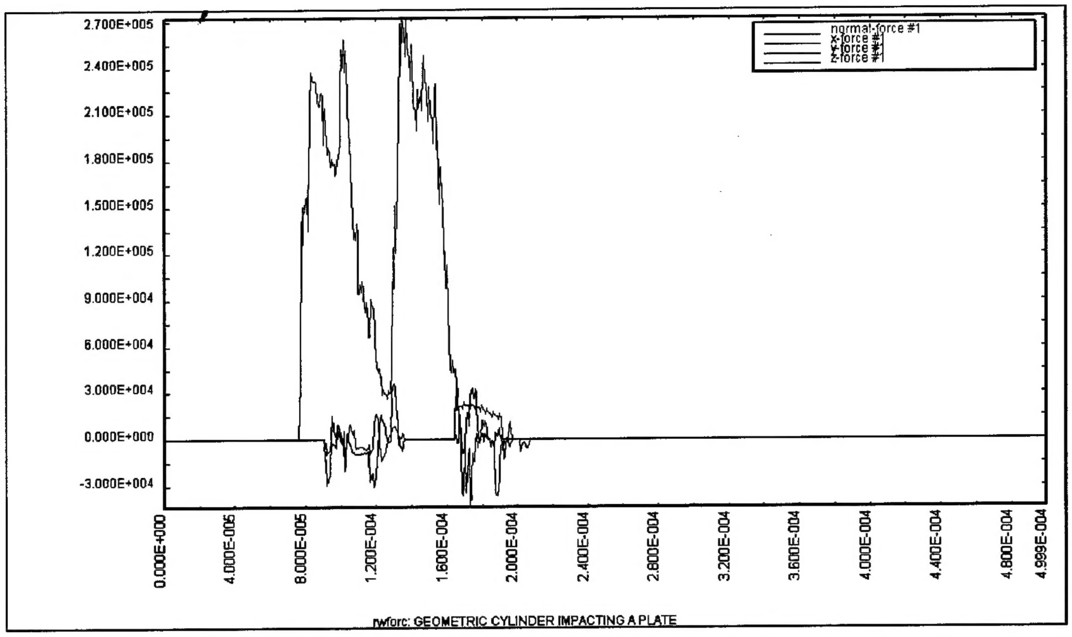

Figure 11. Rigid Wall Force Graph for the Helicopter Crash.

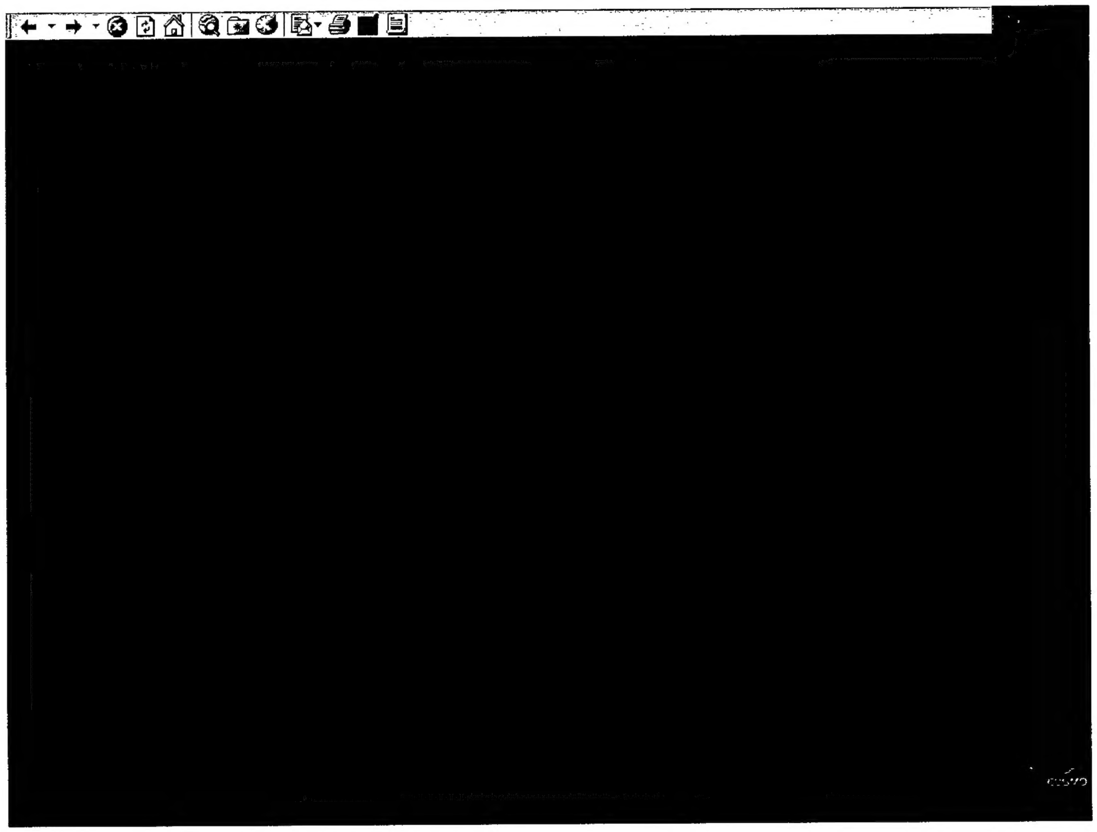

Figure 12. Visualization: Cylindrical Beam Under Compression (Cross Section).

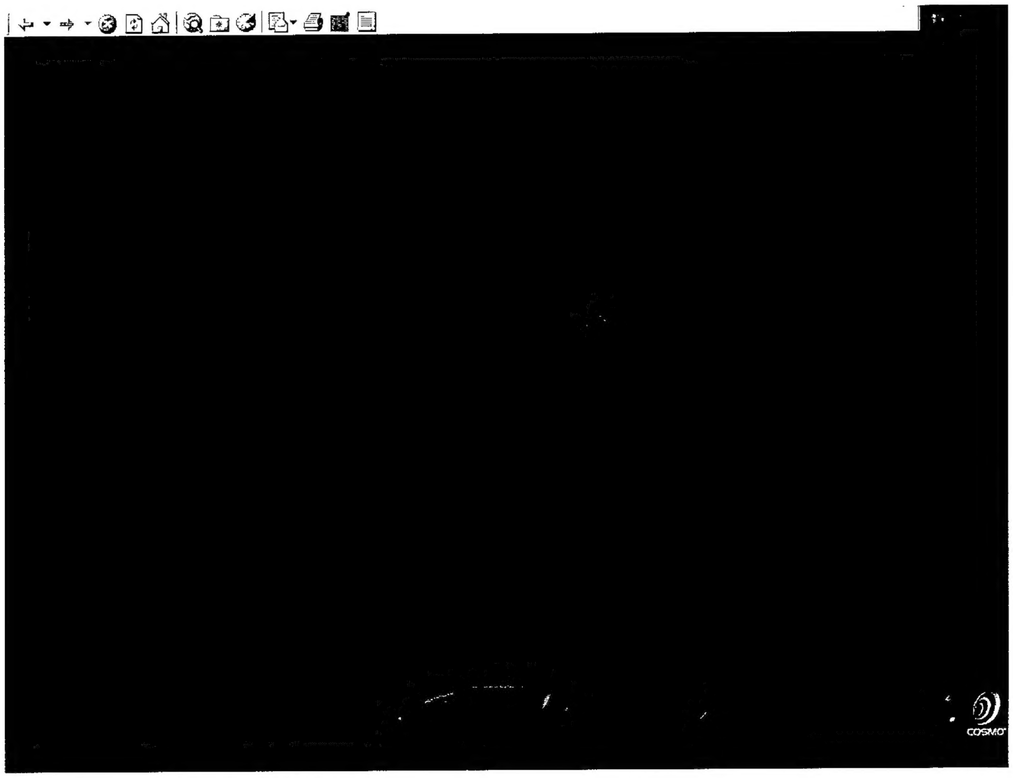

Figure 13. Visualization: Ford Fiesta Under Mine Blast.

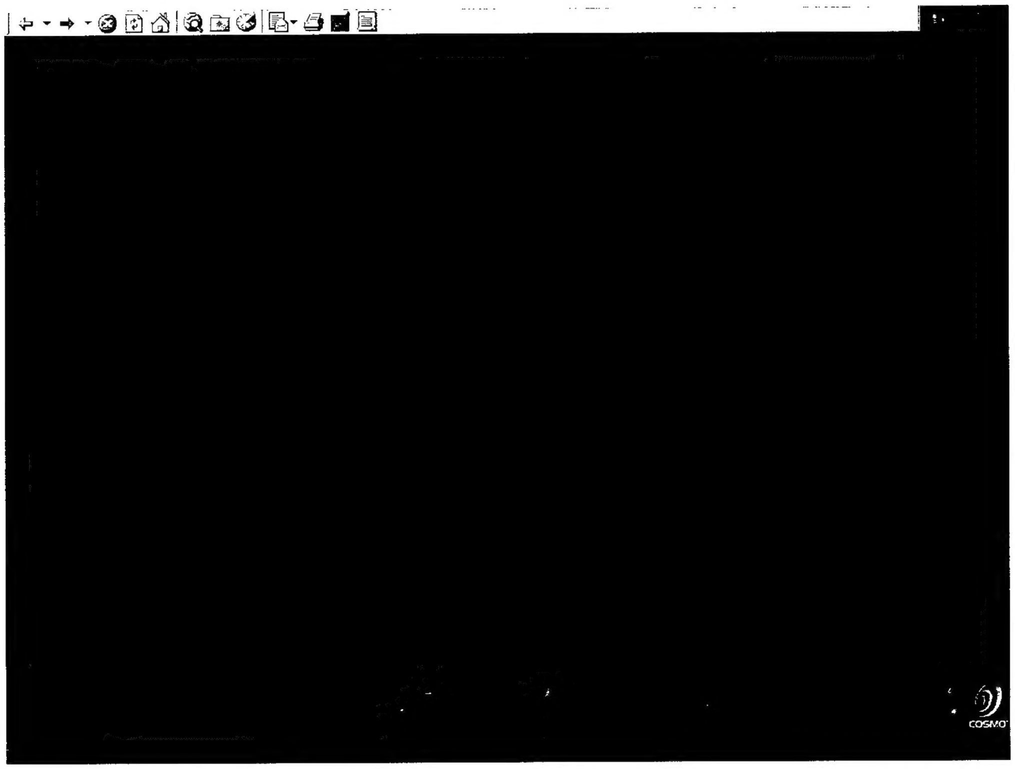

Figure 14. Visualization: Helicopter Model Under Mountain Impact.

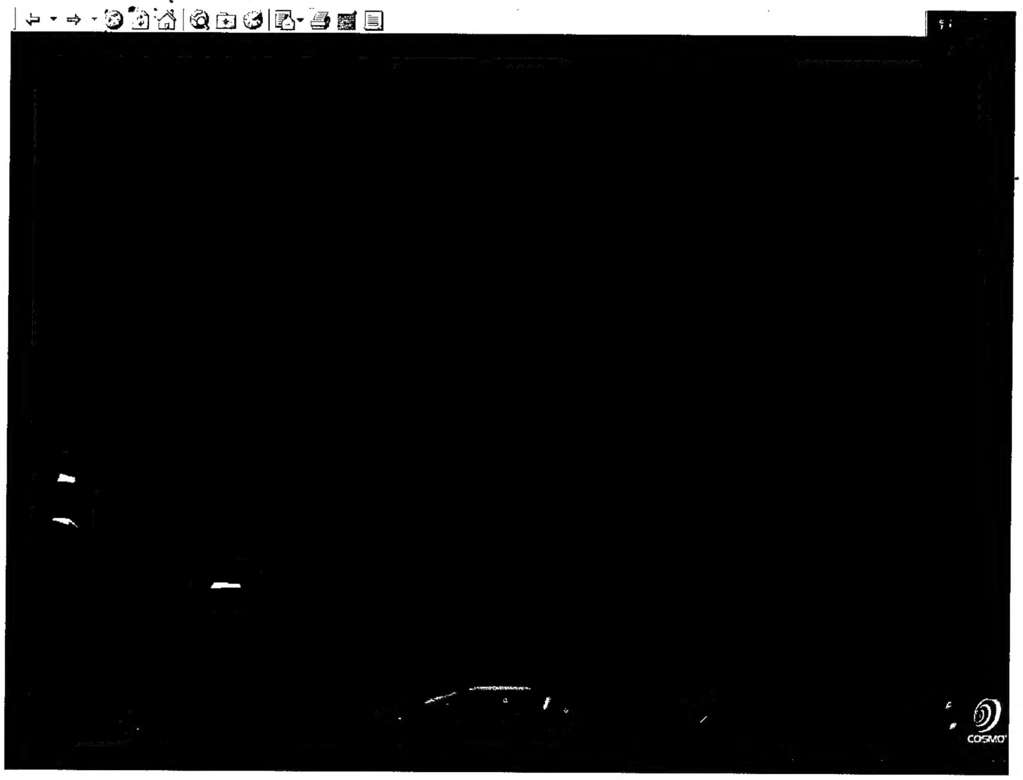

Figure 15. Visualization: Helicopter Model Under Projectile Blast.

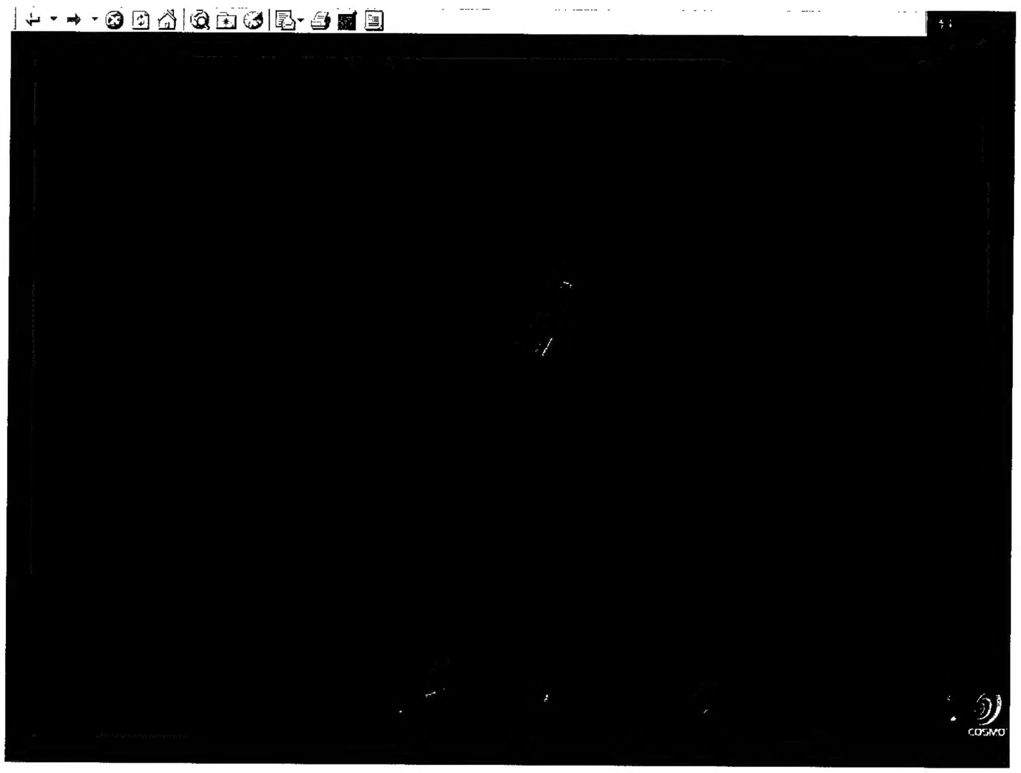

Figure 16. Visualization: Light Truck Under Projectile Blast.